International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 1, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Implementation of SaaS Multitenancy in Cloud Computing

Ms. Sonali S. Kale
Department of Information Technology
Sinhagad Education Society's, SKN COE, Pune, India.
sonali.kale43@gmail.com

Prof. Mr. Ravindra H. Borhade
Department of Information Technology
Sinhagad Education Society's, SKN COE, Pune, India.
ravi_borhade@yahoo.com

***Abstract* — Multitenacy is single web tenant to serve thousands of your customers. In cloud computing environment to implement multitenancy firstly create private cloud and then provide Software-as-a-Service to the customer. The overall aim of this paper is to implement SaaS in cloud computing, to provide service job scheduling for cloud computing, to provide services according to presentation, business and data access layer, to provide security. The main idea of this paper is also to implement the SaaS multitenancy at single instance, so as to reduce the server load.**

Software as a service (SaaS) sometimes referred to as "on-demand software", is a software delivery model in which software and associated data are centrally hosted on the cloud. Multi-tenancy and SaaS are related terms. As, SaaS plus MultiTenancy is equal to SaaS Multi-Tenant, Multi-tenancy is the efficiency level for SaaS. In this paper, mainly result is to provide Software as a Service to customer as per license and provide different web portal and database.

In this ways, Multitenant SAAS helps to create enable a multiple instances of your application and Software as a service provides software on demand of customer. Method of Web technology is used for it.

***Keywords* - SaaS, Multi-Tenant, SaaS Platform.**

I. INTRODUCTION

As, SaaS multitenant architecture is implemented with different technologies. The goal is too continually to manage scalability, portability, customization, security. The key technical challenge is to implement multi-tenancy at single instance. In this paper we will present four major points for enabling multi-tenancy which differ in the degree of resource sharing and development complexity and try to focus on enabling SaaS multitenancy at single instance. We will also present the SaaS framework architecture and their services. Essentially, SaaS platform can be considered a type of specialized application server.

Cloud computing is the use of computing resources like hardware and software that are delivered as a service over a network typically the Internet. The word cloud is used as a metaphor for the Internet, based on the standardized use of a cloud-like shape to denote a network. Cloud is a virtualized pool of computing resources. It can:

1. Manage a variety of different workloads, including the batch of back-end operations and user-oriented interactive applications.
2. Rapidly deploy and increase workload by speedy providing physical machines or virtual machines.
3. Support for redundancy, self-healing and highly scalable programming model, recover from a variety of inevitable hardware/software failure.

Main idea of this paper is the use of cloud for storage and then implementation of SaaS multitenancy for cloud computing environment at single instance. For that purpose we are providing the security considerations too, and then to save time and money we will implement SaaS multitenant architecture at single instance.

II. SAAS SOFTWARE'S ARCHITECTURE

The proposed SaaS platform provides SaaS multitenancy for cloud that serves multiple tenant using a single instance. Tenants can use SaaS services through a web browser.

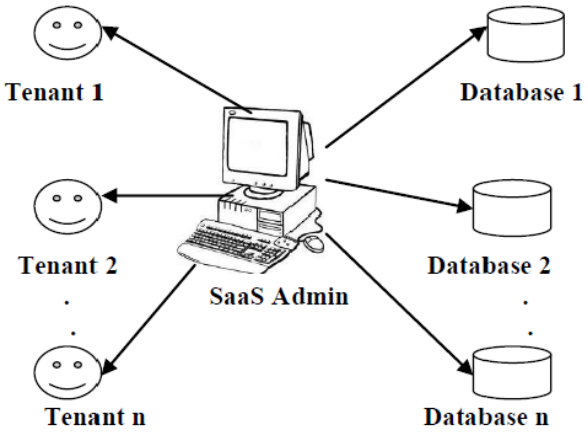

Fig.1. SaaS multitenant architecture for cloud

SaaS multitenant architecture is important to serve the client at single instance. As In proposed system the software web portal is implemented and separate databases are created. So that client can give rent and he will get the license of that software. After providing the SaaS multitenancy for cloud main goal of our paper is to provide services according to these layer given below –

a) Presentation Layer

The presentation layer provides an interface for clients to access the portal application. This layer consists of the following elements:

1. Web Forms**:** The primary web form is the Default.aspx. This page is the entry point to the portal. It is responsible for loading the other elements of the presentation layer.

2. Skins: The Default.aspx web form loads the skin for the page based on the settings for each page or portal.
3. Containers: The Default.aspx web form also loads the containers for the modules based on the settings for each module, page, and portal.
4. Module User Controls: Modules will have at leat a single user control that is the user interface for the module.

b) Business Logic Layer

The business logic layer provides the business logic for all core portal activity. This layer exposes many services to core modules. These services include

1. Localization
2. Caching
3. Exception Management
4. Event Logging
5. Personalization
6. Search
7. Installation & Upgrades
8. Security

The business logic layer is also home to custom business objects that represent most entities that collectively make up the portal. Custom business objects are discussed in more detail later in this chapter. For now it is important to understand that the fundamental purpose of custom business objects is to store information about an object.

c) Data Access Layer

The data access layer provides data services to the business logic layer. This layer allows for data to flow to and from a data store. As described earlier in this chapter, the data access layer uses the Provider Model to allow SaaS Multi-Tenant to support a wide array of data stores. The data access layer consists of two elements:

1. Data Provider API**:** This is an abstract base class that establishes the contract that the implementation of the API must fulfil.
2. Implementation of Data Provider API: This class inherits from the Data Provider API class and fulfils the contract by overriding the necessary members and methods. The core SaaS Multi-Tenant release provides a Microsoft SQL Server implementation of the Data Provider API.

III. SAAS PLATFORM FOR CLOUD

As described above main aim of this paper is to implement the SaaS multitenancy at single instance and to provide services according to different layers. So, for that we first have to build the private cloud and then the independent SaaS platform. So, for that we have to first learn what exactly the responsibilities of SaaS are. So, SaaS platform's responsibilities[8] are

1) Tenancy partitioning
2) Scaling
3) Monitoring & Metering
4) Distributed Services
5) Event log
6) Scheduling

As shown in Figure 2. independent SaaS platform [8] have different users and providers who take services from SaaS platform. SaaS platform lower the technical and financial barriers. SaaS platform provides the application development by providing the ready to use software. Because of these softwares customer get good services.

Imagine you have a web application that you've been selling in the marketplace. You see the writing on the wall and realize that Software as a Service (SaaS) on a cloud infrastructure is the way the industry is headed. You know you need it and your own customers may already be pressuring you to offer a SaaS version for your product. The challenge is to do the conversion to SaaS quickly, efficiently, and in a way that will maintain or enhance your profitability.

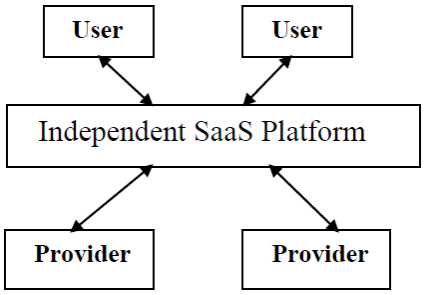

Fig.2. Independent SaaS Platform[8]

There are many differences that must be taken into account for a SaaS application versus a regular web application. Some of these are technical and some are related to the change in business model that a company must adapt to when delivering SaaS.

SaaS multitenancy can be implemented for private or public cloud. As, In this paper we are mostly focusing on private cloud. A private cloud is designed to offer the same features and benefits of public cloud systems, but removes a number of objections to the cloud computing model including control over enterprise and customer data, worries about security, and issues connected to regulatory compliance.

IV. ACHIEVING MULTITENACY FOR CLOUD AND MANAGING MULTITENANT DATA

"Multi-tenant" is short-hand for two key architectural components of a true SaaS solution: 1) single software version and 2) shared infrastructure costs. Both of these have game-changing benefits for both the customer and the SaaS vendor:

1. Single Software Version:

This means that all customers are automatically upgraded to the latest version. All customers not only get the latest innovations and improvements in the software, they also get unparalleled top-to-bottom support and avoid the risk of being orphaned on an old unsupported, unmaintained version. SaaS vendors only need to maintain one (current) version of the software, and every fix/enhancement goes to every customer. Multi-version vendors end up spending 70-90 percent of their development effort simply on maintenance, according to The Economist, as they fragment their maintenance

International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 1, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

efforts over more and more versions of the software and each fix applies to a small subset of customers.

2. Shared Infrastructure:

Multi-tenant allows multiple customers to run on the same infrastructure. Most people think of servers as the main infrastructure cost, but even more costly are the cost of database licenses and other infrastructure software, the 20%-per-year maintenance costs for all infrastructure components, and (most importantly) the operations personnel costs: DBAs, system administrators, network administrators, etc. Fifty customers, each buying and building their own infrastructure, would together spend an order or magnitude more cost than a shared-infrastructure SaaS solution. The customer gains from the SaaS vendor passing along a dramatically lower cost of ownership, and the SaaS provider can also provide capabilities that a single customer couldn't ever justify, such a hot-backup mirror sites, world-class scalability and performance, etc.

A. Achieving Multitenancy

There are three different methods for achieving multitenancy: using a database, using virtualization, or through physical separation. Many researchers consider virtualization to be an alternative to multitenancy. However, virtualization is simply another technology for achieving multitenancy, especially for infrastructure as a service (IaaS). You can achieve virtualization using virtual machine technology that provides a hardware emulation layer over the real hardware. This technology can run multiple copies of server operating systems within one physical machine, and it can share physical hardware, such as network cards and disk storage, between the virtual operating system instances. Such multitenancy virtualization might reduce the costs and expenses of the provided services, but compared to multitenancy using databases technology, virtualization is more costly. First degree multi-tenancy is like Salesforce.com. All parts of the application infrastructure are shared by all customers: the hardware, the application software and server, and the database tables and schema. Technically though even Salesforce does not abide by this definition of multi-tenancy since their database has grown so large that even their underlying Oracle database can no longer support it - it is partitioned into eight identical systems globally, each supporting a different groups of customers. Upgrades and maintenance happens coincidentally across all systems.

Second degree multi-tenancy is where the vendor hosts many clustered multi-tenanted copies of the application. The same application software and data schema are replicated. The advantage of the approach is that rather than spend a lot of money on a single high-horsepower hardware server, many smaller commodity servers can work to service the application. Some vendors who use this approach depart further from the pure SaaS model, by allowing customers to decide when they wish to upgrade. Lesser degree multi-tenancy is an approach advocated by Oracle. Oracle hosts software for customers in a way that is unique to each customer. The schemas for different customers may not be the same because of customizations. Customers still share some components of the server infrastructure so some limited amount of quantities of scale can be achieved. Upgrades in this kind of environment are much more difficult, and often coordinated simultaneous rollouts for all customers is not possible or can be extremely complex.

B. Managing Multitenant Data

A tenant is any application -- either inside or outside the enterprise -- that needs its own secure and exclusive virtual computing environment. This environment can encompass all or some select layers of enterprise architecture, from storage to user interface. All interactive applications (or tenants) have to be multi-user in nature.

Multitenancy means sharing of infrastructure and databases ie. Resources in order to gain price and performance advantages. Multitenancy is not limited to infrastructure, but impacts all layers of abstraction in cloud. Approaches for management of cloud are:

1. Storing tenant data in separate databases, which is the simplest approach to data isolation;
2. Housing multiple tenants in the same database, with each tenant having its own set of tables grouped into a schema created specifically for the tenant;
3. Using the same database and same set of tables to host multiple tenants' data.

Fig.3. illustrates the general architecture for representing multitenancy for effective cloud environments. This architecture employs customer integration on three layers:

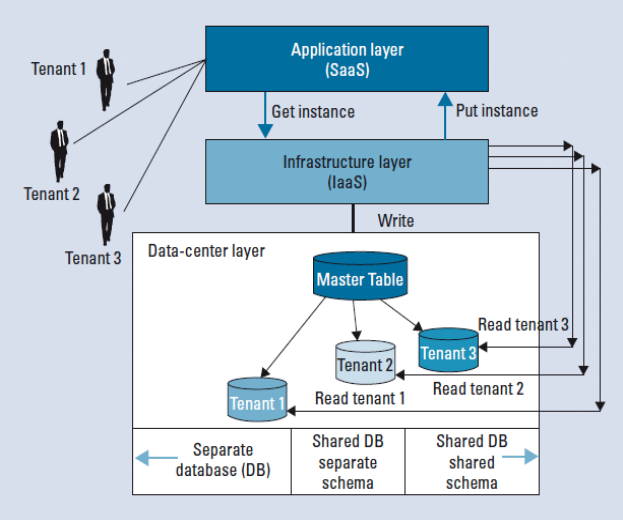

Fig.3. Managing multitenant data

- The application layer,
- The infrastructure layer, and
- The data-center layer.

The data-center-layer multitenancy is well known and provides the highest level of security, if implemented correctly. Infrastructure- and application-layer customer integration for multitenancy are new additions to the cloud computing topology design. The infrastructure layer dedicates one stack of software to a specific customer, deploying stacks for each customer account. The hardware requirements depend on actual service use. For application-layer multitenancy, architectural implementations concern both the software and infrastructure layers.

International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 1, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

V. RELATED WORK AND RESULTS

The overall objective of the project is to implement SaaS multitenancy in Cloud Computing at single instance, create Scheduled Task and to provide services according to layers- Presentation Layer, Business Layer, Data Access Layer. As shown in Fig. 4.

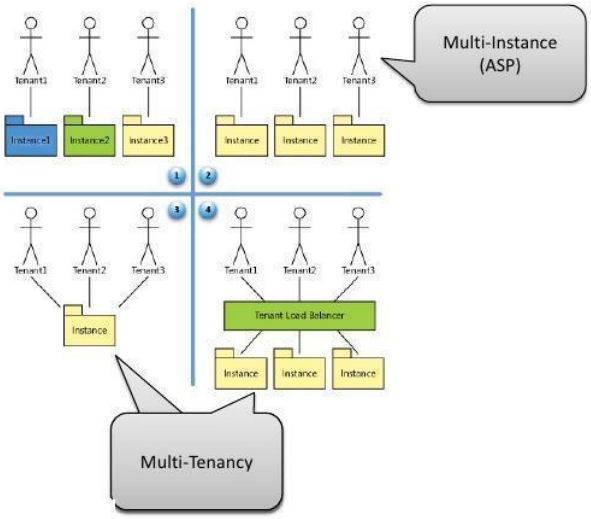

Fig.4. An overview of general multitenancy cloud architecture [11].

In related work we are implementing different features as given:

Single Instance Multi-Tenant: It is an architecture in which a single instance of a software application serves multiple customers.

Template Based Tenant creation **:** Template contains a pre-defined set of pages, components, services etc. that would be automatically created for tenant when they choose one of the available templates like small-scale organization, mid-scale organization, large-level organization.

*Robust Architecture***:** A Robust architecture is the conceptual model that defines the structure, behavior, and more views of a system.

*Modular Architecture for logical components to add/update/Remove***:** It is an upgrading to the next generation design architecture or is that you can replace or add any one component (module) without affecting the rest of the system.

*Tenant files Backup / Restore***:** It is storage of files, resources etc. for specific tenant. Tenant backup is created in order to avoid server load.

Extension for Third party application to integrate: Easily collaboration and provide to integration with external components, resources etc.

Managed/Easy fixes deployments: Easy fixes deployments process is continuous processes of applying quality control — small pieces of effort, applied frequently to improve the quality of software and to reduce the time taken to deliver it, by replacing the traditional practice of applying quality control after completing all development.

Localization: The process of translating a tenant into different languages or adapting a language for a specific country or region.

Authentication and SSO Support: Authentication and Single sign-on (SSO) is a property of access control of multiple related, but independent software systems. With this property a user logs in once and gains access to all systems without being prompted to log in again at each of them.

Able to add new Functional modules without stopping applications: It is providing functionality to deployment of new version without affecting existing application functionality.

*Scalability***:** Scalability is the ability of a system, network, or process, to handle growing amounts of work in a graceful manner or its ability to be enlarged to accommodate that growth.

Easy Framework upgrade: Easy framework upgrade refers to the replacement of a tenant with a newer version of the same tenant. It is most often used in computing and consumer electronics, generally meaning a replacement of hardware, software or firmware with a newer or better version, in order to bring the system up to date or to improve its characteristics. Contrast update and replace.

Easily Extensible: Easy extensibility is a system design principle where the implementation takes into consideration future growth. It is a systemic measure of the ability to extend a system and the level of effort required to implement the extension. Extensions can be through the addition of new functionality or through modification of existing functionality. SaaS is another version of the failed Application Service provider, or ASP (application service provider). SaaS is just a marketing term; in fact, ASP and SaaS are basically the same. The important thing is that now the environment seems to be ready for, and reacting positively on, SaaS developments.

As a result of implementation of SaaS Multitenancy for private cloud we can provider the software on demand to the customer as per license scheme. Tenants can use SaaS services through a web browser. the software web portal is implemented and separate databases are created. So that client can give rent and he will get the license of that software. Then according to templates and different features provided to that web portal given to the customer invoice bill will be generated and deadline will be given to customer for payment. Actually here tenant will be created and then account for the customer is created and different features like metering, event logging, dashboard facility will be provided to the customer.

In this way, different features of SaaS multitenancy are implemented to ensure the clouds storage and then implemented with data security.

VI. CONCLUSIONS

As, Main idea of this paper is the use of cloud for storage and then implementation of SaaS multitenancy for cloud computing environment at single instance. For that purpose we are providing the security considerations too,

and then to save time and money we will implement SaaS multitenant architecture at single instance. So, mainly this idea is been concluded as part of recent trends in industry for acquiring the market situation and then saving time and money of customer.

Cloud computing is not a total new concept; it is originated from the earlier large-scale distributed computing technology. However, it will be a subversion technology and cloud computing will be the third revolution in the IT industry, which represent the development trend of the IT industry from hardware to software, software to services, distributed service to centralized service. Cloud computing is also a new mode of business computing, it will be widely used in the near future. Many companies are currently introducing their SaaS software service platforms to the market. In this paper, The overall objective of the project is to implement SaaS multitenancy in Cloud Computing at single instance, create Scheduled Task and providing the services according to different layers. Customers will get the best software services by paying and getting license.

It is very important to implement SAAS multitenant architecture for cloud either public or private by many of different ways. So here, SaaS multitenancy for private cloud is implemented and event log, metering and service scheduling features are implemented. So that customer can use the best software as service at single instance to save time and money with quality features by getting license.

In this way, conclusion of this paper is to implement SaaS multitenancy for cloud at single instance and to provide different services quickly and then give license of that software to that customer.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

REFERENCES

[1] John W. Rittinghouse, James F. Ransome, “Web Services Delivered from the Cloud”, *Cloud Computing- Implementation, management & security*, Ed. Taylor & Francis Group, LLC , 2010, pp.69-93.

[2] Ralph Mietzner, Tobias Unger, Robert Titze,Frank Leymann, "Combining Different Multi-tenancy Patternsin Service-Oriented Applications," edoc,pp.131-140, 2009 IEEE International Enterprise Distributed Object Computing Conference(edoc2009)

[3] Wei Sun,Kuo Zhang,Shyh-Kwei Chen,Xin Zhang,HaiqiL iang, Software as a Service:An Integration Perspective,Proceedings of the5th international conference on Service-Oriented Computing, September17-20,2007,Vienna,Austria

[4] Hai Henry, Sakoda, Seitaro, “SaaS and integration best practices", F David Banks, John S. EricksonMichael Rhodes, “Multi-tenancy in Cloud-based Collaboration Services”, Hewlett-Packard Development Company, L.P., February 21, 2009 itsu Scientific and Technical Journal, v 45, n 3, p. 257-264, July 2010

[5] Manish Godse, Shrikant Mulik, "An Approach for Selecting Software-as-a-Service (SaaS) Product," cloud

[6] Cor-Paul Bezemer, Andy Zaidman, “Multi-Tenant SaaS Applications: Maintenance Dream or Nightmare?”, *Delft University of Technology Software Engineering Research Group Technical Report Series*, 2010, http://www.se.ewi.tidelft.nl /techreports, pp 1-5.

[7] Jihyun Lee Sung Jin Hur ,”*Level 2 SaaS platform and platform management framework*” Advanced Communication Technology (ICACT), 2011 13th International Conference.

[8] Guoling Liu ,Sch. of Inf. Sci. & Technol., Shandong Inst. of Light Ind., Jinan, China ,”*Research on Independent SAAS platform*” ,Information Management and Engineering (ICIME), 2010 The 2nd IEEE International Conference on 16-18 April 2010

[9] Jinan Fiaidhi , Irena Bojanova ,Jia Zhang ,Liang-Jie Zhang , “Enforcing Multitenancy for Cloud Computing Environments” January/February 2012 (Vol. 14, No. 1) pp. 16-18 1520-9202/12/$31.00 © 2012 IEEE

[10] Sungjoo Kang, Sungwon Kang, Sungjin Hur,”*A Design of the Conceptual Architecture for a Multitenant SaaS Application*” 2011 First ACIS/JNU International Conference on Computers, Networks, Systems, and Industrial Engineering.

[11] http://www.slideshare.net/rstropek/saas-multitenancy-and-cloud-computing.

AUTHOR’S PROFILE

Ms. Sonali S. Kale

Birth place: Ahemadnagar DOB: 4th March 1986. Received Bachelor degree of Information Technology from SRES College of engineering, Kopargaon under affiliation of Pune University (2007) & pursuing Post graduate degree in Information Technology from STES, SKN College of engineering, Pune under affiliation of Pune University, having 2 years industry and 4 years of teaching experience. Research areas of interests are Cloud computing, Database, Networking. Email-Id: sonali.kale43@gmail.com

Prof. Mr. Ravindra H. Borhade

Received Bachelor(1999) and Postgraduate degree (2001) in Electronics and telecommunication engineering from College of engineering, Pune under Pune University & pursuing PHD from Pune university, having 2 years Industry and 11 years of teaching experience and working as Head of Department in STES, SKN College of engineering, Pune. Research areas of interest are Wireless communication, Networking. Email-Id: ravi_borhade@yahoo.com